



FIG. 1

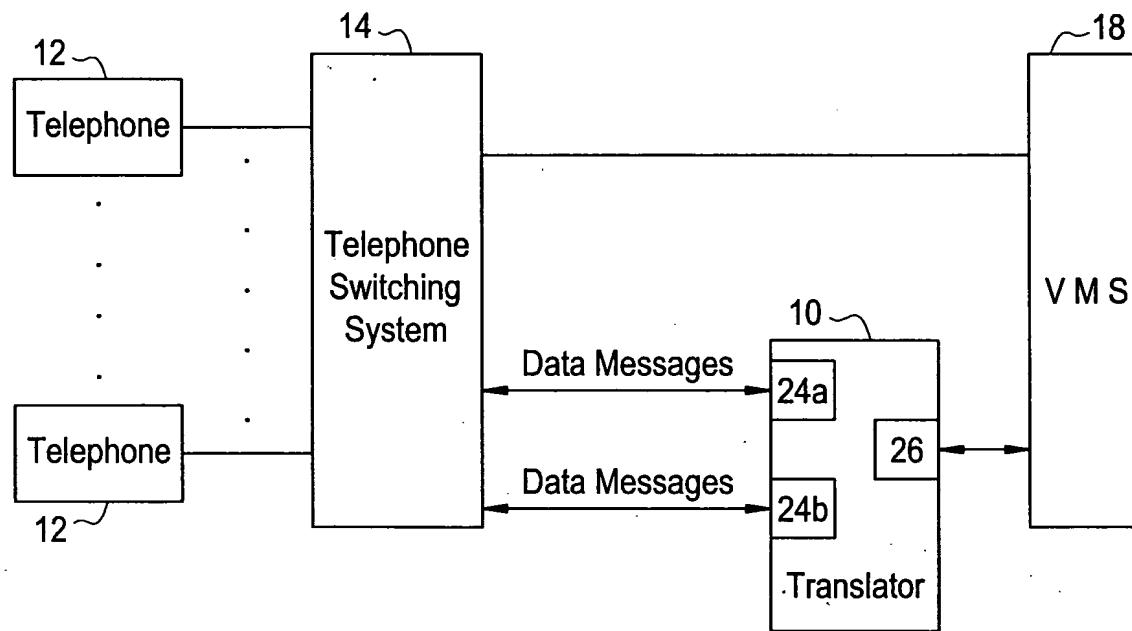


FIG. 2A

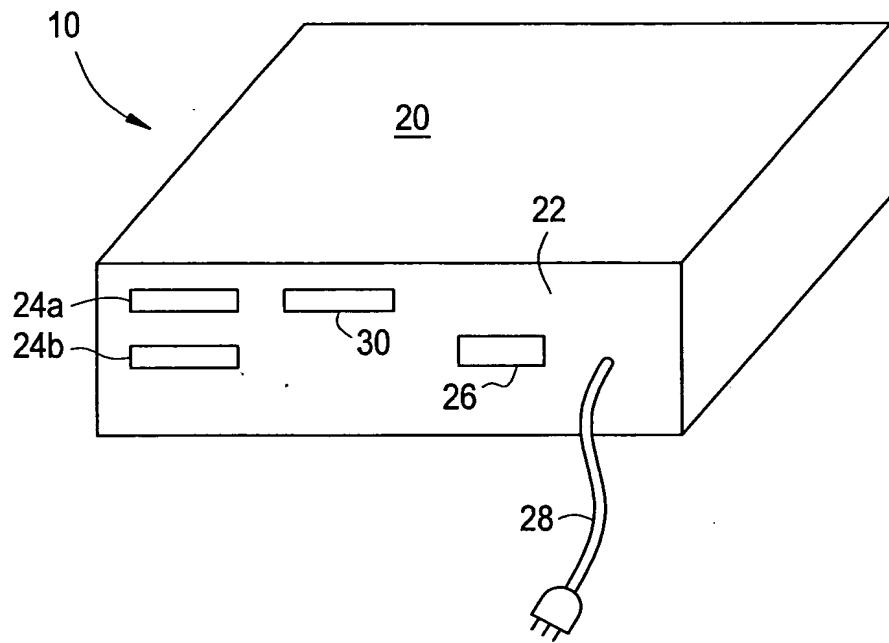


FIG. 2B

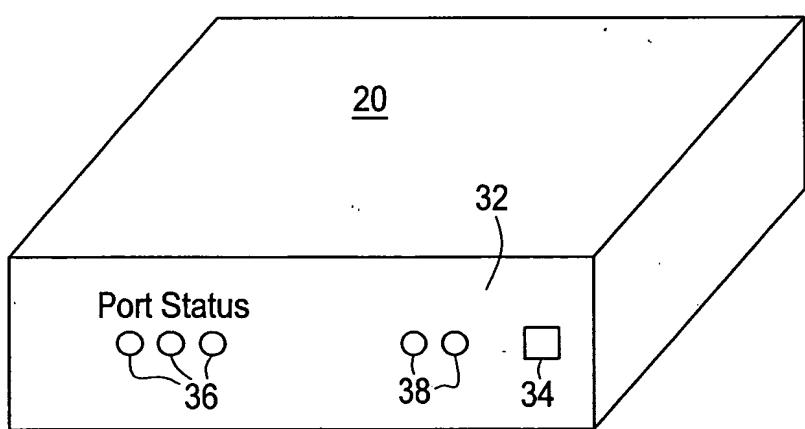
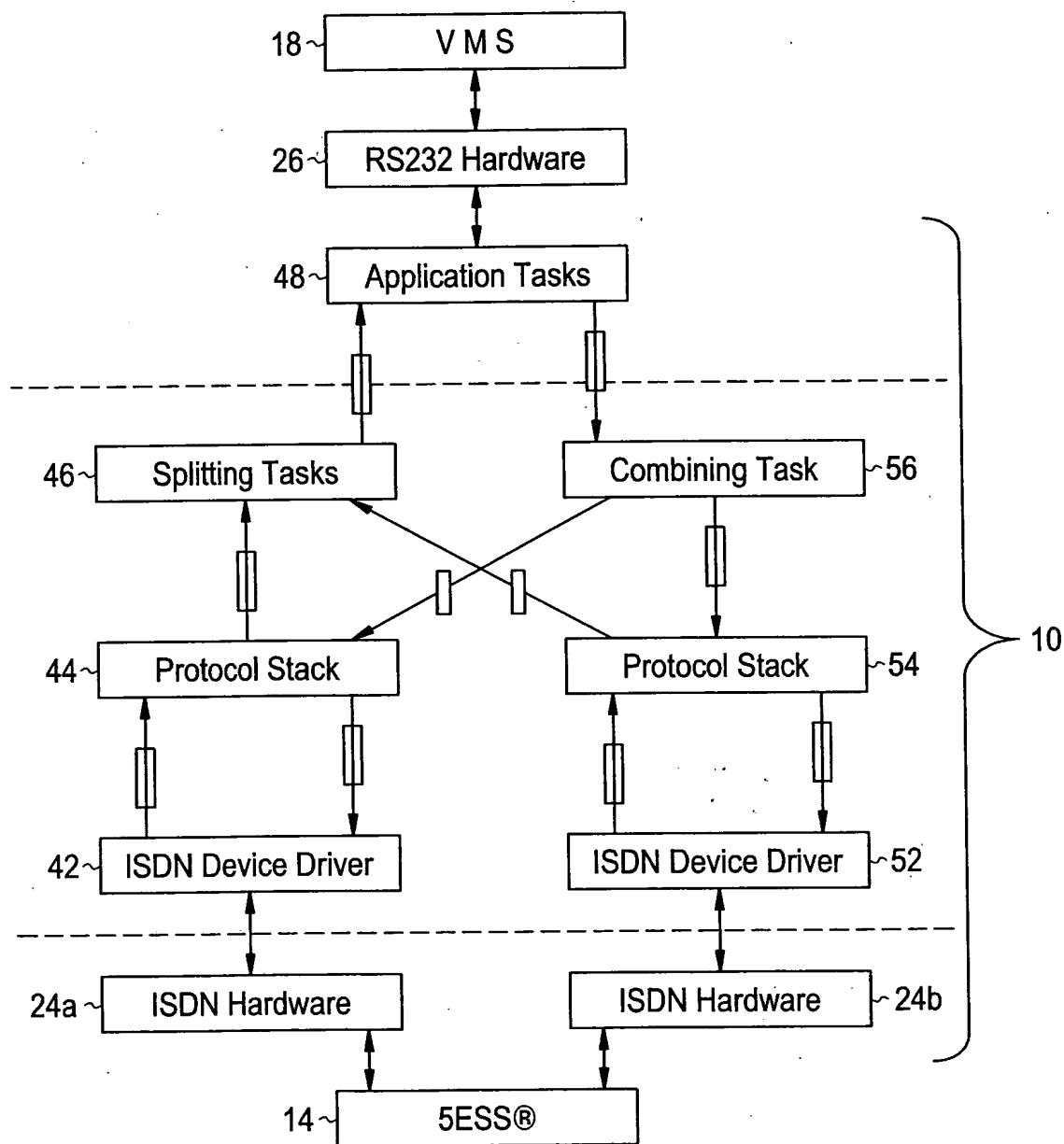


FIG. 3



## FIG. 4A

### ISDN DRIVER (Interrupt Service Routine)

If the Cause of Interrupt IS A New frame received

    Receive the frame from 5ESS switch through the hardware in a temporary buffer

    Test for its integrity

    If it is a good frame Forward it to the Protocol Stack, else discard it

End If

If the Cause of Interrupt IS Link Deactivated

    Reset the protocol stack

    Link\_Deactivated=TRUE

End If

If the Cause of Interrupt IS Link Activated AND Link\_Deactivated=TRUE

    Link\_Activated=FALSE

    Power up the protocol stack

End If

## FIG. 4B

### PROTOCOL STACK

The protocol stack consists of three sub-layers each implementing the data link, network and transportation layer of the protocol. Each layer does the following processing

FOREVER

1. Wait for a 5E Frame.
2. Determine if the frame is a control packet or a data packet.
3. If it is a control packet
  - a. Send an appropriate response (varies for error Control, flow control, link integrity checking etc.,) to the lower layer.
  - b. Based on the type of the control packet, take the stack to a different state.
4. If the received packet is a data packet without errors AND the stack is in Data Transfer State, forward it to the upper layer of the stack. (Upper layer for the three sub-layers will be network layer, transportation layer and splitting task).

END FOREVER

**REPLACEMENT  
SHEET**

## **FIG. 4C**

**SPLITTING TASK:**

**FOREVER**

1. Read the message from the protocol stack.
2. Split the message into individual call messages.
3. Forward it to the application task.

**END FOREVER**

## **FIG. 4D**

**APPLICATION TASK:**

**FOREVER**

1. Read the message delivered by the Splitting Task.
2. Translate the message from the 5ESS format to 1AESS format.
3. Deliver it to the VMS through the RS232 Hardware.

**END FOREVER**

## FIG. 4E

### APPLICATION TASK:

FOREVER

1. Receive the message from the VMS through the RS232 hardware.
2. Translate the message from the 1AESS format to the 5ESS format.
3. Forward it to the COMBINING TASK.

END FOREVER

## FIG. 4F

### COMBINING TASK:

LinkToBeSent=1

FOREVER

Receive The Message from the Application and Combine Them

If both the links are in a working condition

Send the Combined Message to the stack corresponding to Link indicated by LinkToBeSent

If LinkToBeSent=0 then LinkToBeSent=1 else LinkToBeSent=0

End if

If one of the links is failed, send it to the stack corresponding to the healthy link.

End FOREVER

## FIG. 4G

### PROTOCOL STACK

Stands Blocked until the protocol stack is in the DATA TRANSFER STATE.

FOREVER

Wait for a Message from the COMBINING TASK

Receive in a temporary buffer and Append Control Information to the translated message.

Forward it to the ISDN Device Driver

END FOREVER

## FIG. 4H

### ISDN DRIVER (Interrupt Service routine)

IF the cause of the Interrupt is "Ready for Transmission"

If the input message queue has any message

Transmit it to the 5ESS Switch through the hardware

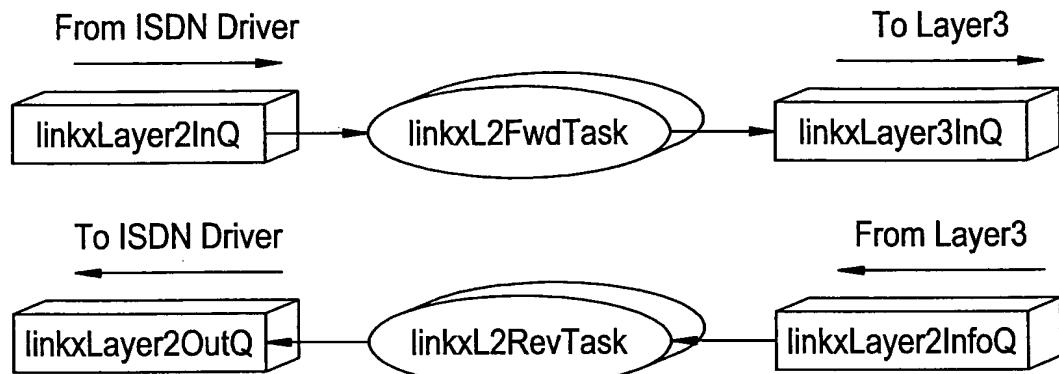
End If

End If

# APPENDIX A1

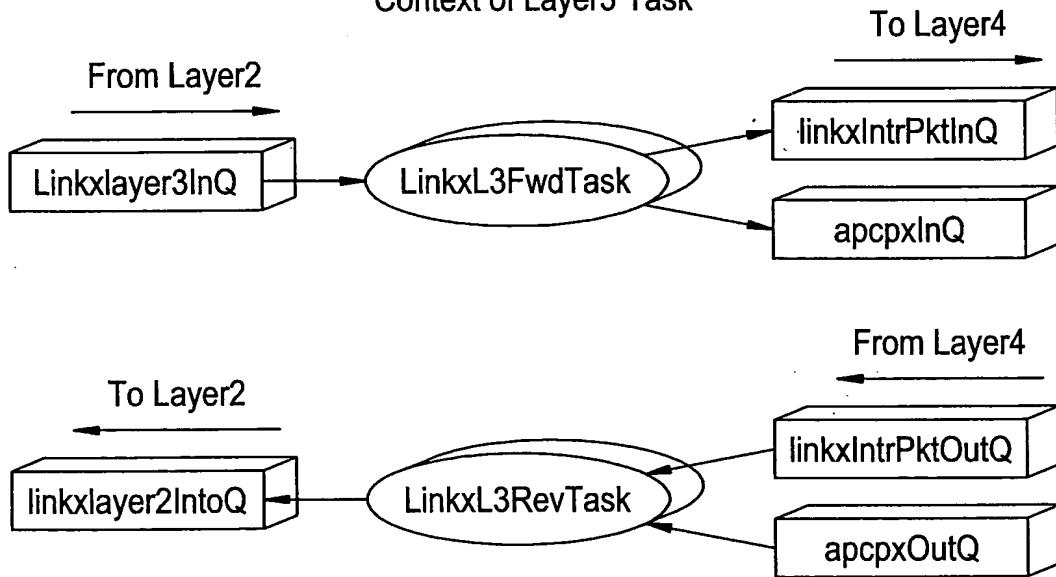
## Task Context Diagrams

### Context of Layer2 Task



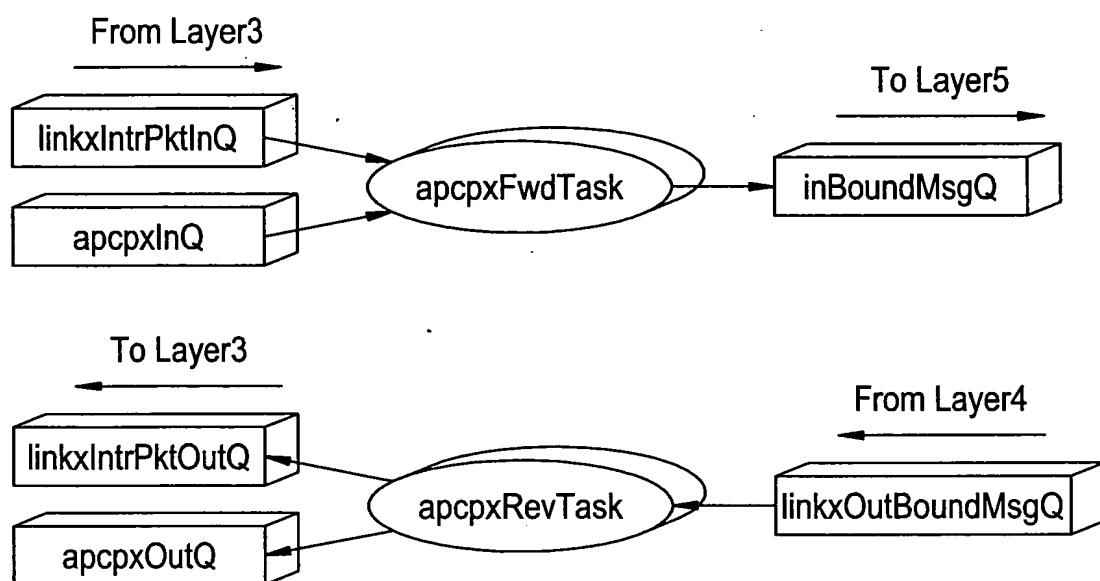
# APPENDIX A2

### Context of Layer3 Task



## APPENDIX A3

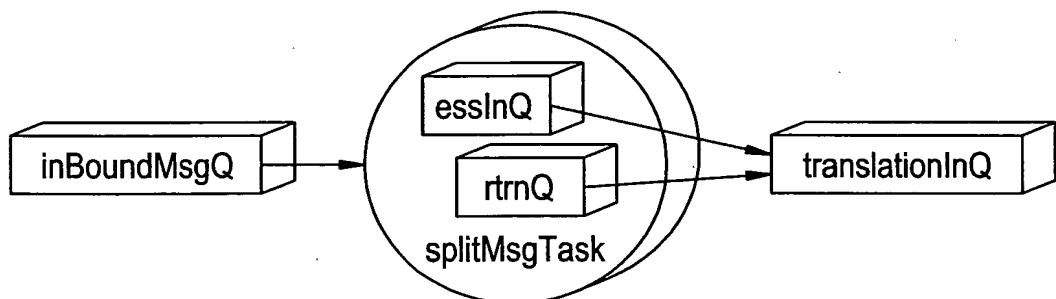
### Context of APCP Task



Interrupt task context has to be included here.

## APPENDIX A4

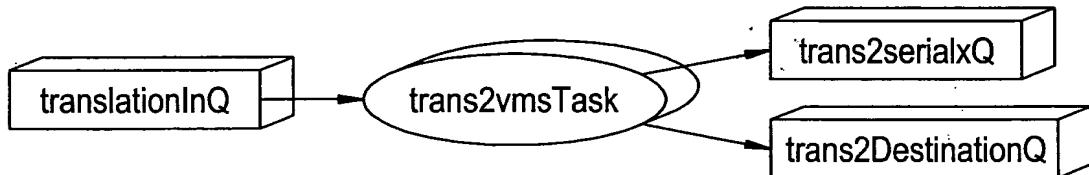
Context of splitMsgTask:



Context of grpMsgTask:



Context of trans2vmsTask:

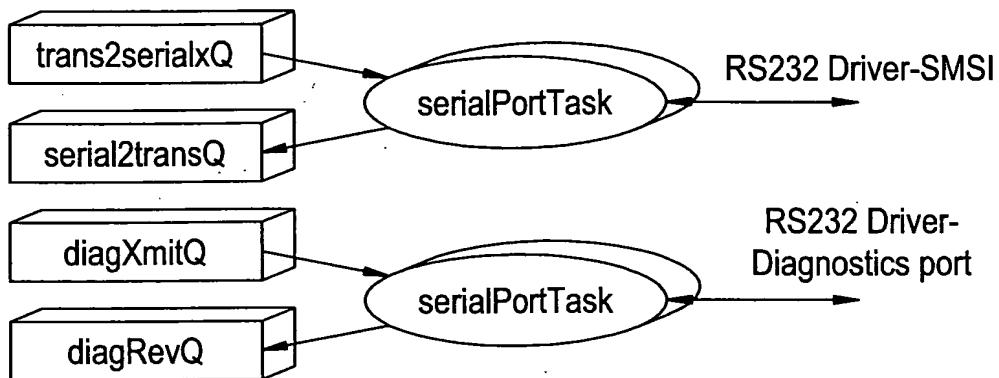


Context of vms2transTask:



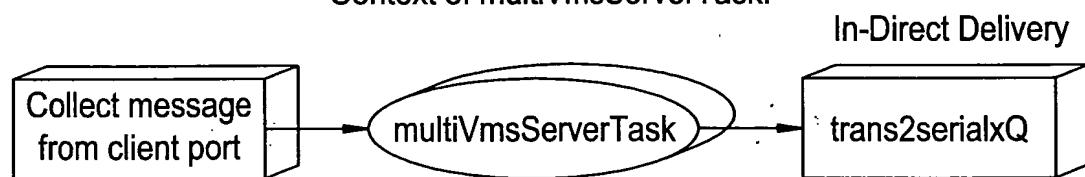
## APPENDIX A5

Context of serialPortTask:



## APPENDIX A6

Context of multiVmsServerTask:



Context of multiVmsClientTask:

